AD-A037 141   FEDERAL COBOL COMPILER TESTING SERVICE WASHINGTON D C   F/G 9/2
COBOL COMPILER VALIDATION SUMMARY REPORT.(U)
FEB 77
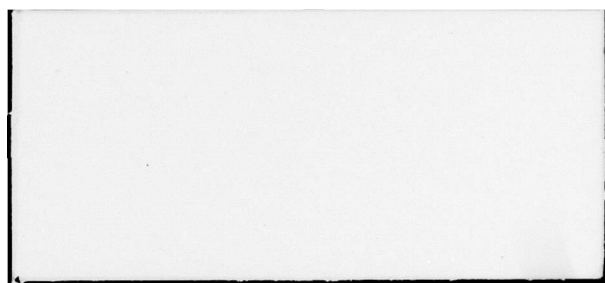UNCLASSIFIED   CCVS74-VSR175   NL

| OF |
AD
A037141

END
DATE
FILMED
4—77

(10)

(6) COBOL COMPILER
VALIDATION SUMMARY REPORT.

(14) VALIDATION NUMBER CCVS74-VSR175

(11) 17 Feb 77

(12) 52 p.

Prepared By:

FEDERAL COBOL COMPILER TESTING SERVICE
DEPARTMENT OF THE NAVY (AFPESO)
WASHINGTON, D.C. 20376

D D C
RECEIVED
MAR 21 1977
A

408438

BIBLIOGRAPHIC DATA SHEET	1. Report No. CCVS74-VSR175	2.	3. Recipient's Accession No.

4. Title and Subtitle	5. Report Date
Validation Summary Report #CCVS74-VSR175 IPI BLIS/COBOL Version 2.0	17 February 1977
	6.

7. Author(s) Same as organization - see 9.	8. Performing Organization Rept. No.

9. Performing Organization Name and Address	10. Project/Task/Work Unit No.
Federal COBOL Compiler Testing Service ADPE Selection Office Department of the Navy Washington, D. C. 20376	11. Contract/Grant No.

12. Sponsoring Organization Name and Address	13. Type of Report & Period Covered
Automatic Data Processing Equipment Selection Office Department of the Navy Washington D. C. 20376	14.

15. Supplementary Notes

16. Abstracts

This Validation Summary Report (VSR) for the Information Processing Incorporated COBOL Compiler Version 2.0 provides a consolidated summary of the results obtained from the validation of the subject compiler against the 1974 COBOL Standard (X3.23-1974/FIPS PUB 21-1). The compiler was validated at the Low level of FIPS PUB 21-1. The VSR is made up of several sections showing the discrepancies found. These include an overview of the validation which lists all categories of discrepancies by level/module within X3.23-1974, a section relating the categories of discrepancies to each of the Federal levels of the language; and a detailed listing of discrepancies together with the tests which were failed.

17. Key Words and Document Analysis. 17a. Descriptors

Programming Languages
Standards
COBOL
Compilers
Verifying
Proving Program Correctness
Software Engineering
FIPS PUB 21-1

17b. Identifiers/Open-Ended Terms

CCVS
CVS

17c. COSATI Field/Group   09/02

18. Availability Statement	19. Security Class (This Report) UNCLASSIFIED	21. No. of Pages 50
Release Unlimited.	20. Security Class (This Page) UNCLASSIFIED	22. Price

FORM NTIS-35 (REV. 3-72)   **THIS FORM MAY BE REPRODUCED**   USCOMM-DC 14952-P72

CCVS74-VSR175

COBOL COMPILER VALIDATION

1. Validation Number   CCVS74-VSR175

2. Hardware Vendor   Data General Corp.

3. Software Vendor   Information Processing Inc. (IPI)

4. Mainframe   MM1 NOVA 1200 Compatible

5. Compiler Identification   IPI BLIS/COBOL — Version 2.0

6. Operating System Identification   IPI BLIS/COBOL — Version 2.0

7. Compiler Validation System Version Number   CCVS74 1.0

8. Federal Information Processing   21-1
       Standard Publication

*PLEASE NOTE. The Federal COBOL Compiler Testing Service may make full
and free public disclosure of the Validation Summary Report (VSR) in
accordance with the "Freedom of Information Act" (5 U.S.C. #552). The
results of this validation are only for the purpose of satisfying United
States Government requirements, and apply only to the Computer System,
Operating System release, and compiler version identified in the VSR. The
COBOL Compiler Validation System is used to determine, insofar as is
practical, the degree to which the subject compiler conforms to the Fed-
eral COBOL Standard. Thus, the VSR is necessarily discretionary and judg-
mental. The United States Government does not represent or warrant that
the statements, or any one of them, set forth in the VSR are accurate or
complete. The VSR is not meant to be used for the purpose of publicizing
the findings summarized therein.

For information concerning this compiler you can contact the vendor's
designated representative named below:

        Mr. Robert F. Roycroft
        Senior Consultant
        Information Processing Incorporated
        5237 Edgewater Drive
        Orlando, Florida 32810

2

CCVS74-VSR:75

TABLE OF CONTENTS

3

SECTION 1. INTRODUCTION

1.1 Purpose of the Validation Summary Report

The purpose of the Validation Summary Report (VSR) is to identify individual
COBOL language elements whose implementation does not conform to Federal Stan-
dard COBOL as adopted from American National Standard Programming Language
COBOL, X3.23-1974, by Federal Information Processing Standard 21-1 (FIPS
PUB 21-1).

1.2 Preparation of the VSR

The Validation Summary Report is prepared by analyzing the results of running
the COBOL Compiler Validation System (CCVS). The COBOL Compiler Validation
System consists of audit routines containing features of Federal Standard COBOL,
their related data, and an executive routine (VP-routine) which prepares the
audit routines for compilation. Each audit routine is a COBOL program which
includes many tests and supporting procedures indicating the result of the
tests.

The testing of a compiler in a particular hardware/operating system environment
is accomplished by compiling and executing each audit routine. The report pro-
duced by each routine tells whether the compiler passed or failed the tests in
the routine. If the compiler rejects some language elements by terminating
compilation, giving fatal diagnostic messages, or terminating execution abnor-
mally, then the test containing the code the compiler was unable to process is
deleted and the audit routine compilation and execution repeated.

The compilation listings and the output reports of the audit routines con-
stitute the raw data from which the members of the Federal COBOL Compiler
Testing Service produce a Validation Summary Report.

1.3 Organization of the VSR

The Validation Summary Report is made up of several sections the contents of
which are described below.

    a. Section 2 summarizes the results of the compilation and execution
of the programs comprising the COBOL Compiler Validation System. Section 2
is subdivided into a subsection representing each level of each module defined
in American National Standard Programming Language COBOL, X3.23-1974. Each
subsection contains a list of all of the language elements which must be
implemented in order to claim support of that level/module. The list of
language elements will be annotated to include a description of both syntax
and semantic errors detected during the validation.

    b. Section 3 - FIPS PUB 21-1 defines four Federal levels of the COBOL
Standard. Section 3 of the VSR lists the discrepancies described in Section 2
by the Federal level in which the problem occurs.

4

c. Section 4 contains information which describes the software environment in which the compiler was tested. This includes the name and version of the operating system; the implementor-names which were used in the Environment Division of the programs comprising the CCVS; the options used with the compiler; and if applicable, information regarding the use of compiler optimization features.

d. Appendix A is the Validation Summary Working Document, a working paper resulting from the compilation and execution of the CCVS, and from which the VSR is derived.

1.4 Abstract Covering Compliance to American National Standard Programming Language COBOL

Definition of an Implementation of American National Standard Programming Language COBOL (excerpts from X3.23-1974, Chapter 1, Section 1.5).

An implementation is defined to meet the requirements of the American National Standard COBOL specification if that implementation includes a fully implemented specified level of each of the functional processing modules and of the Nucleus as defined in this Standard. It follows from this that, in order to meet the requirements of this Standard, an implementation must:

a. Not require the inclusion of substitute or additional language elements in the source program, in order to accomplish any part of the function of any of the standard language elements.

b. Accept all standard language elements contained in a given level of a module which is specified as being included in the implementation, except as specifically exempted (as pertaining to specific hardware components for which support is not claimed). See "Elements that Pertain to Specific Hardware Components" below.

These points are of particular pertinence in two areas:

(1) There are throughout the American National Standard COBOL specification certain language elements whose syntax, or effect, is specified to be, in part, implementor-defined. While the implementor specifies the constraints on that portion of each element's syntax or rules that is indicated in this Standard to be implementor-defined, such constraints may not include any requirement for the inclusion in the source program of substitute or additional language elements.

(2) When a function is provided outside the source program that accomplishes a function specified by any particular standard COBOL element, then the implementation must not require, except for Environment Division elements, the specification of that external function in place of or in addition to that standard language element:

The following qualifications apply to the American National Standard COBOL specification:

5

a. There are certain language elements which pertain to specific types of hardware components. In order for an implementation to meet the requirements of this standard, the implementor must specify the minimum hardware configuration required for that implementation and the hardware components that it supports. Further, when support is thus claimed for a specific hardware component, all standard language elements that pertain to that component must be implemented if the module in which they appear is included in the implementation. Language elements that pertain to specific hardware components for which support is not claimed, need not be implemented. However, the absence of such elements from an implementation of American National Standard COBOL must be specified.

b. An implementation of American National Standard COBOL may include the ENTER statement or not, at the option of the implementor.

c. An implementation that includes, in addition to a specified level of each of the functional processing modules and of the Nucleus, elements or functions that either are not defined in the American National Standard COBOL specification or are defined in a given level of a standard module not otherwise included in the implementation, meets the requirements of this Standard. This is true even though it may imply the extension of the list of reserved words by the implementor, and prevent proper compilation of some programs that meet the requirements of this Standard. The implementor must specify any optional language (language not defined in a specified level but defined elsewhere in the Standard) or extensions (language elements or functions not defined in this Standard) that are included in the implementation.

d. In general, the American National Standard COBOL specification specifies no upper limit on such things as the number of statements in a program, the number of operands permitted in certain statements, etc. It is recognized that these limits will vary from one implementation of American National Standard COBOL to another and may prevent the proper compilation of some programs that meet the requirements of this standard.

IMPLEMENTOR-DEFINED LANGUAGE SPECIFICATIONS

The language elements in the following lists depend on implementor definitions to complete the specification of the syntax or rules for the elements.

The elements whose syntax is partly implementor-defined are:

Element	Implementor-Defined Aspect
SOURCE-COMPUTER paragraph	computer-name
OBJECT-COMPUTER paragraph	computer-name
MEMORY SIZE clause	integer
alphabet-name	implementor-name; whether imple-

5

	mentor-names are provided.
SPECIAL-NAMES paragraph	implementor-name
ASSIGN clause	implementor-name
VALUE OF clause	implementor-name; whether implementor-names are provided.
RERUN clause	implementor-name and the form; the implementor provides at least one of seven specified forms.
CALL and CANCEL statements	relationship between operand and the referenced program.
COPY statement	relationship between library-name text-name, and the library.
ENTER statement	language-name
Margin R	The location.
Area B	The number of character positions.
Qualification	The number of qualifiers; at least five must be supported.

The elements whose effect is partly implementor-defined are:

Element	Implementor-Defined Aspect
alphabet-name	The correspondence between native and foreign character sets.
implementor-name switches	Whether setting can change during execution.
USAGE IS COMPUTATIONAL clause	Representation and whether automatic alignment occurs.
USAGE IS INDEX clause	Representation and whether automatic alignment occurs.
SYNCHRONIZED clause	Whether implicit FILLER positions are generated; their effect on the size of group items and redefining items.
ACCEPT statement	Maximum size of one transfer of data in Level 1 Nucleus.

7

DISPLAY statement   Maximum size of one transfer of data
                                     in Level 1 Nucleus.

Numeric test   Representation of valid sign in the
                                     absence of the SIGN IS SEPARATE
                                     clause.

Comparison of nonnumeric items   Collating sequence, where NATIVE or
                                     implementor-name collating sequence
                                     is implicitly or explicitly
                                     specified.

Arithmetic expressions   Number of places carried for inter-
                                     mediate results.

Elements That Pertain to Specific Hardware Components

The standard language elements in the list that follows pertain to specific
types of hardware components. These language elements must be implemented
in an implementation of American National Standard COBOL when support is
claimed, by the implementor, for the specific types of hardware components to
which they pertain, and the module in which they are defined is included in
that implementation.

Element   Hardware Component
-------   ------------------

CODE-SET clause   Device capable of supporting the
                                     specified code.

MULTIPLE FILE TAPE clause   Reel

CLOSE...REEL/UNIT statement   Reel or mass storage

CLOSE...NO REWIND statement   Reel or mass storage

OPEN...REVERSED statement   Reel with the capability of making
                                     records available in the reversed
                                     order; mass-storage with the cap-
                                     ability of making records avail-
                                     able in the reversed order.

OPEN...NO REWIND statement   Reel or mass storage

OPEN...I-O statement   Mass storage
  (Sequential I-O only)

OPEN EXTEND statement   Reel or mass storage

REWRITE statement   Mass storage
  (Sequential I-O only)

8

CCVS74-VSR175

SEND...BEFORE/AFTER ADVANCING statement	Devices capable of vertical positioning; devices capable of action based on mnemonic-names.
USE...I-O (Sequential I-O only)	Mass storage
WRITE...BEFORE/AFTER ADVANCING	Devices capable of vertical positioning; devices capable of action based on mnemonic-name.

1.5   The Federal COBOL Standard

The COBOL compiler validation results enclosed in this document reflect the
degree to which the subject COBOL compiler implements the Federal COBOL
Standard. The Federal COBOL Standard is essentially the same as the
American National Standard Programming Language COBOL, X3.23-1974, with two
exceptions:

> The Federal COBOL Standard defines 4 levels and the ANSI
> Standard defines only the minimum COBOL implementation and
> the full standard. Low and High levels of the Federal
> COBOL Standard (see 1.5.1) correspond to the above two ANSI
> levels (minus the Report Writer module). Two additional
> levels, low-intermediate and high-intermediate have been
> included in the Federal Standard between the highest and
> lowest subsets. These additional levels accommodate
> hardware which cannot support the full standard, but which
> is capable of implementing more than the minimum standard.

> The Report Writer Module is not contained in the Federal
> COBOL Standard.

The Federal COBOL Standard requires that a compiler contain as a minimum the
elements specified in at least one of the Federal levels. No restrictions are
imposed on the inclusion of selected features from higher levels or even unique
vendor extensions. Compatibility among various implementations of a given
level containing additional features must be controlled by management imposed
standards and restrictions.

1.5.1   Federal Standard COBOL Levels

     a.   Federal Standard COBOL specifications are the language specifications
contained in American National Standard Programming Language COBOL, X3.23-1974.
For purposes of the Federal Standard, the modules defined in X3.23-1974 are
combined into four levels. Not all computers are large enough to accommodate a
COBOL compiler containing the full ANSI Standard. Therefore, the Federal
Government requires that all compilers acquired by its agencies contain as a
minimum one of the four Federal levels, depending on machine size, configuration
and user needs. The knowledge that all computers will support at least one of
these four subsets simplifies the task of developing machine-independent COBOL
programs.

     b.   The four levels of Federal Standard COBOL are identified as: Low,
Low-Intermediate, High-Intermediate, and High. Each Federal Standard COBOL
level is composed of either the high or low levels of the nucleus and ten of
the eleven Functional Processing Modules (FPMs) defined in X3.23-1974.
The four Federal Standard COBOL levels are reflected in the following table.
The numbers in the table refer to the level within the FPM or nucleus as
designated in X3.23-1974, and a dash in the table denotes that the corres-
ponding FPM is omitted.

CCVS74-VSR175

	Low Level	Low Inter- mediate Level	High Inter- mediate Level	High Level
NUCLEUS	1	1	2	2
FPMs				
TABLE HANDLING	1	1	2	2
SEQUENTIAL I-O	1	1	2	2
RELATIVE I-O	-	1	2	2
INDEXED I-O	-	-	-	2
SORT-MERGE	-	-	1	2
REPORT WRITER	-	-	-	-
SEGMENTATION	-	1	1	2
LIBRARY	-	1	1	2
DEBUG	-	1	2	2
INTER-PROGRAM COMMUNICATION	-	1	2	2
COMMUNICATION	-	-	2	2

1.5.2 Conformance to Federal Standard COBOL

A compiler implemented in conformance to Federal Standard COBOL must meet at least the following requirements.

    a. The implementation must include all of the language elements of at least one of the levels of Federal Standard COBOL.

    b. The implementation must meet all of the requirements defined in American National Standard COBOL, X3.23-1974, Section I, paragraph 1.5, Definition of An Implementation of American National Standard COBOL which is provided in section 1.4 of this VSR.

    c. The implementation must provide a facility for the user to optionally specify a level of Federal Standard COBOL for monitoring his source program at compile time. The monitoring will be an analysis of the syntax used in a source program against the syntax included in the specified level of Federal Standard COBOL. Any syntax used in the source program that does not conform to that allowed by the user selected level of Federal Standard COBOL will be diagnosed. The syntax diagnosed as not conforming to the specified level will be identified to the user through a diagnostic message on the source program listing. The diagnostic message will contain, at least: (1) The identification of the source program line number in which the nonconforming syntax occurs, (2) the identification of the level of Federal Standard COBOL that supports

11

the syntax or that the syntax is nonstandard COBOL.

1.6. Use of the VSR

The Federal COBOL Compiler Testing Service may make full and free public disclosure of the Validation Summary Report (VSR) in accordance with the "Freedom of Information Act" (5 U.S.C. #552). The results of the validation are only for the purpose of satisfying United States Government requirements, and apply only to the computer system, operating system release, and compiler version identified in the VSR.

The COBOL Compiler Validation System is used to determine, insofar as is practical, the degree to which the subject compiler conforms to the COBOL Standard. Thus, the VSR is necessarily discretionary and judgmental. The United States Government does not represent or warrant that the statements, or any one of them, set forth in the VSR are accurate or complete. The VSR is not meant to be used for the purpose of publicizing the findings summarized therein.

1.7 Sources of Additional Information

FIPS PUB 21-1 defines the Federal COBOL Language Standard. This publication is available from the Office of ADP Standards Management, National Bureau of Standards, Washington, D. C., 20234.

The detailed COBOL language specifications are given in the publication "American National Standard Programming Language COBOL, X3.23-1974", available from the American National Standards Institute, 1430 Broadway, New York, New York 10018.

An explanation of the COBOL Compiler Validation System is contained in the CCVS User's Guide. This document explains how to run the compiler validation system. The User's Guide and a magnetic tape containing a copy of the CCVS programs are available from the National Technical Information Service, Springfield, Virginia, 22151. (Ordering information can be obtained from the Federal COBOL Compiler Testing Service.)

1.8. Requests for Interpretation

Questions regarding this VSR or the CCVS in general should be forwarded to the FCCTS. If any problem cannot be adequately resolved through the FCCTS, the request for interpretation will be forwarded to the Federal COBOL Interpretation Committee for final resolution.

A brochure describing the Validatvon process including the procedures for requesting a validation and resolution of questions involving interpretation of the current Federal Standard is available from the Department of the Navy, Federal COBOL Compiler Testing Service, Washington, D.C. 20376.

SECTION 2.   DETAILED EVALUATION OF ERRORS.

This section summarizes the results of the compilation and execution of the
programs comprising the COBOL Compiler Validation System (CCVS). The version
of the CCVS used during this validation is shown inside the front cover of the
VSR.

Section 2 is made up of a variable number of subsections. The number of sub-
sections is dependent on the Level of Federal COBOL being validated. There will
be a subsection for each level of each module which is validated. If the high
level of a module is validated then there will be two subsections for that
module; one for the low level and one for the high level.

A validation of the low level of Federal Standard COBOL would result in three
subsections being present. One for Nucleus level 1, one for Sequential I-O
level 1, and one for Table Handling level 1.

Each error or deviation noted in this section makes reference to a program or
functional COBOL module contained in Appendix A (Validation Summary Working
Document). This reference provides the documented results of an occurrence
of errors/deviataions detected during the running of the CCVS using the compiler
within the environment identified within this document. The Validation Summary
Working Document is presented in sequence by functional module, functional
module level and program number as defined below.

Each program in the COBOL Compiler Validation System is identified by a 5-
character program name. The name associates the routine with the functional
processing module and level of American National Standard Programming Language
COBOL tested within the program.

The five character name has the general format XXNMM. The first two
characters are alphabetic and identify the functional module tested by
the program. The permissable values are:

        CM - Communication
        DB - Debug
        IC - Inter-Program Communication
        IX - Indexed I-O
        LB - Library
        NC - Nucleus
        RL - Relative I-O
        RP - Report Writer
        SG - Segmentation
        SQ - Sequential I-O
        ST - Sort-Merge
        TH - Table Handling

The third character of the audit routine name is either a 1 or 2, and
identifies the level of the functional module being tested. Each module and
level is represented by several programs. The fourth and fifth characters of
the program name are sequence numbers for programs which test features in the

13

same level of the same functional processing module.

As an example, the program name NC210 is the tenth program in the series of routines which test the second level of the Nucleus module.

Description of Section 2.

Each error/deviation is noted by number in the left hand margin opposite the language element in question. This number is used in section 3 to categorize errors by Federal level (See 1.5.1). Inserted directly below the language element is a brief description of the error. To the right of the language element is a page reference to X3.23-1974, American National Standard Programming Language COBOL. The reference at the end of the description of the error is to Appendix A which contains the detailed information collected during the validation. The reference is made up of the routine name followed by an A or B (A for compile time or syntax error and B for execution time or semantic error) and a number which makes the error unique in Appendix A.

Example:

2.1 Nucleus Level 1

.
.
.

     Operational symbols: S V P   II-21
2.1.9   --
      * The scaling character 'P' is not permitted in a
      * PICTURE character-string.
      *   (NC101.A.2)

.
.
.

2.2 Sequential I-O Level 1

2.1.9 represents the ninth error for Nucleus Level 1

II-21 represents the page in X3.23-1974 where the language
     element is defined

*   Boxes the description of the error/deviation

NC101.A.2 represents:

    Program name - NC101
    Syntax error - A
    second error - 2

CCVS74-VSR175

2.1 NUCLEUS LEVEL 1

    2.1.1   ---
            *   Only the first 12 positions of procedure-names and
            *   data-names are used by this compiler; therefore, these
            *   names must be unique within the first 12 characters.
            *   (Refer to Nucleus Module paragraph 5.3.2.2.1 on page
            *   I-76.)
            *   (NC104 8.3)

    2.1.2   ---
            *   The compiler does not accept procedure-names composed
            *   entirely of the numeric characters (0-9). (Refer to
            *   Nucleus Module paragraph 5.3.2.2.1.1.3 Paragraph-name

                                  16

CCVS74-VSR175

2.1.3   --
    *   The compiler does not accept the syntax of other than
    *   01 level Data Division entries which begin in Area A.
    *   (Refer to Reference Format paragraph 5.8.4 on page
    *   I-108.)
    *   (NC113 A)

2.1.4   --
    *   The compiler does not accept the syntax of certain
    *   procedural statements which either end in column 72
    *   of the current COBOL source image or the statement
    *   is completed on the next line. (Refer to paragraph
    *   5.8.3.3 on page I-107.)
    *   (NC107 A.7 and 1H105 A.3)

CCVS74-VSR175

2.1.5
----
    *   The compiler does not accept mnemonic-names and
    *   condition-names for switches in the SPECIAL-NAMES
    *   paragraph. (Refer to Nucleus Module paragraph 3.1.3.2
    *   on page II-5.)
    *
                                          (NC105 A)
----

    alphabet-name clause
2.1.6
----
    *   The compiler does not accept the syntax of the alphabet-
    *   name clause in the SPECIAL-NAMES paragraph. (Refer to
    *   Nucleus Module paragraph 3.1.3.2 on page II-8.)
    *   (NC114 A.1)

    CURRENCY SIGN clause
2.1.7
----
    *   The compiler does not accept PICTURE character-string
    *   symbols in the Data Division that were defined in the
    *   CURRENCY SIGN clause of the SPECIAL-NAMES paragraph.
    *   (Refer to Nucleus Module paragraph 3.1.3.4(6) on page
    *   II-10.)
    *   (NC107 A.1)

    DECIMAL-POINT clause
2.1.8
----
    *   The compiler does not accept the reversal of decimal

18

```
*     point and commas for PICTURE character-string symbols
*     in the Data Division as defined in the DECIMAL-POINT
*     IS COMMA clause of the SPECIAL-NAMES paragraph.  (Refer
*     to Nucleus Module paragraph 3.1.3.4(7) on page II-10.)
*                                              (NC107 A.2)
```

2.1.9
```
*     The compiler rejected an alphanumeric literal used to
*     initialize a data item described with a BLANK WHEN
*     ZERO clause.  (Refer to Nucleus Module paragraph
*     4.3.3(2) on page II-14.)
*                                              (NC107 A.3)
```

2.1.10
```
*     The compiler does not accept the syntax for the PICTURE
*     character-string X(26)BX(12)BX(10) indicating the
*     character-string exceeded 30 characters.  (Refer to
*     Nucleus Module paragraph 4.7.2(3) on page II-19.)
*                                              (NC105 A.1)
```

2.1.11
```
*     The compiler does not recognize the scaling position
*     character "P" in the PICTURE character-string.  (Refer
*     to Nucleus Module paragraph 4.7.4(5) on page II-20.)
*                                              (NC101 A.3)
```

```

CCVS74-VSR175

2.1.16  ---------------------------------------------------------------
        *  .   An incorrect answer was produced by an ADD identifier/
        *      literal series which included a mixture of integer and
        *      noninteger values.  (Refer to Nucleus Module paragraph
        *      5.5.4(2) on page II-56.)
        *                                            (NC106 B.2)
        ---------------------------------------------------------------
        TO identifier
        GIVING identifier
        ROUNDED phrase
        SIZE ERROR phrase
2.1.17  ---------------------------------------------------------------
        *      The ON SIZE ERROR path was not taken and the resultant
        *      identifier was modified when a size error condition
        *      occurred on the ADD statement.  (Refer to Nucleus Module
        *      paragraph 5.3.2 on page II-50.)
        *                                            (NC106 B.1)
        ---------------------------------------------------------------

2.1.18  ---------------------------------------------------------------
        *      The compiler does not support the ALTER statement.
        *      (Refer to Nucleus Module paragraph 5.6 on page II-57.)
        *                                        (NC102 A.1)
        ---------------------------------------------------------------

2.1.19  ---------------------------------------------------------------
        *      The compiler rejected the ROUNDED phrase of the DIVIDE
        *      statement and produced incorrect results when the
        *      composite of operands equaled 18 digits.
        *                               (NC117 A.3 and NC117 B.5
        ---------------------------------------------------------------
        SIZE ERROR phrase
2.1.20  ---------------------------------------------------------------
        *      The ON SIZE ERROR path was not taken and the resultant-
        *      identifier was modified when a size error condition
        *      occurred on the DIVIDE statement.  (Refer to Nucleus
        *      Module paragraph 5.3.2 on page II-50.)
        *                                    (NC117 B.1, 2, & 3)
        ---------------------------------------------------------------

21

2.1.21

    *   The compiler does not support the INSPECT statement.
    *   (Refer to Nucleus Module paragraph 5.14 on page II-68.)
    *                                    (NC115 A)

2.1.22

    *   The move of an alphanumeric elementary item to a numeric
    *   elementary item did not follow the rules for an elementary
    *   MOVE.  (Refer to Nucleus Module paragraph 5.14.4(4)b and
    *   5.14.4(6) on page II-75 and II-76.)
    *                                  (NC104 B.2)

2.1.23

    *   The execution of a test of the MULTIPLY statement caused
    *   the audit routine to abnormally terminate.  (Refer to
    *   Nucleus Module paragraph 5.16 on page II-77.)
    *                                  (NC101 B)

2.1.24

    *   The ON SIZE ERROR path was not taken and the resultant
    *   identifier was modified when a size error condition
    *   occurred on the MULTIPLY statement.  (Refer to Nucleus
    *   Module paragraph 5.3.2 on page II-50.)
    *                                  (NC120 B.1)

2.1.25

    *   A PERFORM statement nested 20 levels deep failed to

```
 *   complete execution. (Refer to the Nucleus Module
 *   paragraph 5.17.4(7) on page II-84.)
 *   (NC102 B.2)
----
     procedure-name
     THRU phrase
     TIMES phrase
2.1.26 ---
 *   The compiler does not accept the syntax for the
 *   "identifier TIMES" phrase when the data description
 *   entry for the identifier includes a "S" symbol (sign)
 *   in the PICTURE character string. (Refer to Nucleus
 *   Module paragraph 5.17.4(6)b on page II-80.)
 *   (NC102 A.2)

The STOP statement II-85
     literal
     RUN
The SUBTRACT statement II-89
     identifier/literal series
2.1.27 ---
 *   An incorrect answer was produced by a SUBTRACT
 *   identifier/literal series which included a mixture of
 *   integer and noninteger values. (Refer to Nucleus
 *   Module paragraph 5.20.4(2) on page II-90.)
 *   (NC106 B.5)

     FROM identifier
     GIVING identifier
     ROUNDED phrase
     SIZE ERROR phrase
2.1.28 ---
 *   Several syntax messages were produced on SUBTRACT
 *   statements which included both the ROUNDED and ON SIZE
 *   ERROR phrases.
 *   (NC106 A.2)
2.1.29 ---
 *   The ON SIZE ERROR path was not taken and the resultant-
 *   identifier was modified when a size error condition
 *   occurred on the SUBTRACT statement. (Refer to Nucleus
 *   Module paragraph 5.3.2 on page II-50.)
 *   (NC106 B.4)

```

23

CCVS74-VSR175

2.2   SEQUENTIAL I-O LEVEL 1

2.2.1   ------------------------------------------------------------
        *     The compiler does not support the RERUN clause in the
        *     I-O-CONTROL paragraph.  (Refer to Sequential Module
        *     paragraph 2.1.3.1 on page IV-6.)
        *                                             (SQ113 A)
        ------------------------------------------------------------
2.2.2   ------------------------------------------------------------
        *     The compiler does not support the CODE-SET clause when
        *     alphabet-name is defined in the SPECIAL-NAMES paragraph
        *     (Refer to Sequential Module paragraph 3.5 on page IV-12.)
        *                                             (SQ111 A)
        ------------------------------------------------------------
2.2.3   ------------------------------------------------------------
        *     The compiler does not support the STANDARD option of
        *     the LABEL RECORDS clause.  (Refer to Sequential Module
        *     paragraph 3.7 on page IV-14.)
        *                                             (SQ102 A)
        ------------------------------------------------------------

CCVS74-VSR175

2.2.4    --------------------------------------------------------
    *    A program terminated execution in a test exercising
    *    the REWRITE statement.  (Refer to Sequential Module
    *    paragraph 4.4 on page IV-31.)
    *                                          (SQ115 B)
         --------------------------------------------------------

2.2.5    --------------------------------------------------------
    *    The compiler does not support the DECLARATIVE section
    *    or USE statement.  (Refer to Procedure Division para-
    *    graph 5.7.1.1 on page I-99 and Nucleus Module paragraph
    *    4.5 on page IV-32.)
    *                                          (SQ105 A)
         --------------------------------------------------------

2.2.6    --------------------------------------------------------
    *    Line overprinting did not occur for the statement
    *    WRITE ... AFTER 0 LINE.  (Refer to Sequential Module
    *    paragraph 4.6.4(9)b on page IV-35.)
    *                                          (SQ101 B)
         --------------------------------------------------------

CCVS74-VSR175

2.3  TABLE HANDLING LEVEL 1

Language Concepts

2.3.1
----------------------------------------------------------
*    A syntax message was issued for a subscript that was
*    signed.  (Refer to Subscripting paragraph 5.3.3.8.2
*    on page I-89.)
*                                    (IH105 A.1)
----------------------------------------------------------

2.3.2
----------------------------------------------------------
*    The compiler does not support the Indexing features
*    USAGE IS INDEX, INDEXED BY or SET statement.
*                                    (IH101 A)
----------------------------------------------------------

Data Division

Procedure Division

26

SECTION 3.   COMPILER STATUS

Section 1.5 explains the four levels of Federal Standard COBOL.  This
section lists the discrepancies described in Section 2 by the Federal level
in which the problem occurs.  All errors listed for a lower level are also
errors in any higher level, even though they are listed only in the lower
level.  The paragraph number from Section 2 is used to reference the errors
in each Federal level.

3.1.  Low Level

| | | |
|---|---|---|
| 2.1.1 | - | Names are limited to 12 characters. |
| 2.1.2 | - | Procedure-names must not be entirely numeric characters. |
| 2.1.3 | - | Entries other than 01 level are not accepted in Area A. |
| 2.1.4 | - | Procedure statements ending in column 72 are not accepted. |
| 2.1.5 | - | Switches are not supported. |
| 2.1.6 | - | Alphabet-name clause is not supported. |
| 2.1.7 | - | CURRENCY SIGN clause does not function correctly. |
| 2.1.8 | - | DECIMAL-POINT clause does not function correctly. |
| 2.1.9 | - | Data item described with BLANK WHEN ZERO could not be initialized with VALUE clause. |
| 2.1.10 | - | PICTURES with large character strings are not accepted. |
| 2.1.11 | - | PICTURE scaling position character "P" is not supported. |
| 2.1.12 | - | Redefinition of 77 level entries is not permitted. |
| 2.1.13 | - | SIGN IS TRAILING SEPARATE CHARACTER is only option implemented. |
| 2.1.14 | - | RIGHT option of SYNCHRONIZED clause is not accepted. |
| 2.1.15 | - | Decimal point in VALUE IS clause is not recognized. |
| 2.1.16 | - | ADD identifier/literal series produced an incorrect answer. |
| 2.1.17 | - | ON SIZE ERROR of ADD statement did not function correctly. |
| 2.1.18 | - | ALTER statement is not supported |
| 2.1.19 | - | ROUNDED phrase on a DIVIDE statement was rejected. |
| 2.1.20 | - | ON SIZE ERROR of DIVIDE did not function correctly. |
| 2.1.21 | - | INSPECT statement is not supported. |
| 2.1.22 | - | Move of Alphanumeric  to numeric data item was incorrect. |
| 2.1.23 | - | Program abnormally terminated when testing the MULTIPLY. |
| 2.1.24 | - | ON SIZE ERROR of MULTIPLY statement functioned incorrectly. |
| 2.1.25 | - | PERFORM nested 20 levels failed to complete execution. |
| 2.1.26 | - | Sign in identifier of PERFORM is not permitted. |
| 2.1.27 | - | SUBTRACT identifier/literal series produced an incorrect answer. |
| 2.1.28 | - | SUBTRACT statement with ON SIZE and ROUNDED was not accepted. |
| 2.1.29 | - | ON SIZE ERROR of DIVIDE statement functioned incorrectly. |
| 2.2.1 | - | RERUN clause is not supported. |
| 2.2.2 | - | CODE-SET clause with alphabet-name phrase is not supported. |
| 2.2.3 | - | LABEL RECORD STANDARD is not supported. |
| 2.2.4 | - | Program terminated when testing the REWRITE. |
| 2.2.5 | - | DECLARATIVE and USE features are not supported. |
| 2.2.6 | - | Overprinting did not occur for WRITE .. AFTER 0 LINES. |
| 2.3.1 | - | Subscripts cannot be signed. |
| 2.3.2 | - | Indexing feature of the TABLE HANDLING Module is not |

27

implemented.

3.2. Low-Intermediate Level

Not tested.

3.3. High-Intermediate

Not tested.

3.4. High Level

Not tested.

SECTION 4    SOFTWARE  ENVIRONMENT.

    The compiler referenced in this document was validated using the software
environment described in this section.  When using a modification of the
described environment, the compiler may or may not continue to conform to the
Standard.  It should be noted that during the validation process, an attempt
is made to validate as many different options as possible.

    The use of compiler options, implementor-names in the Environment Division
and any form of optimization which is not described in this report could cause
the compiler to produce a program that does not perform according to the
specifications of Standard COBOL.  Only the environment described in this
document has been used with this compiler to satisfy the requirements of FIPS
PUB 21-1 and FPMR 101-32.1305.1a.  (Any deviations which must be corrected as
per the referenced FPMR are described in Sections 2 and 3 of this report.)

1.  Options or parameters used on the processor call statement for the compiler:
The following options/parameters were used during the validation.

Options specified:  None.

Options defaulted:  None.


2.  Environment Division implementor-names.


    Printer destined files

        PRINTER

    Tape files

        XXXXXNNN.      Implementor-name  was defined at execution-time.

    Sequential Mass-storage files

        XXXXXNNN.      Implementor-name was defined at execution-time.

    Random Access files

        N/A

    Sort files (SD)

        N/A

    Switch names

      Not supported by the compiler.


29

Source Computer names

    XXXXX082.        (No name indicated by BLIS/COBOL Programmer's Manual)

Object Computer names

    XXXXX083.        (No name indicated by BLIS/COBOL Programmer's Manual)

3.  Optimization.  The compiler may or may not have optimization features.
If there was an optimization feature available, it was used during the
validation process (during a separate execution of the Compiler Validation
System) to determine if its use causes the compiler to produce a program which
does not give the expected results.  If the optimization is invoked through
the compiler call statement then it is mentioned in paragraph 1 above.  If
it is invoked through the introduction of syntax in other than the Data
and Procedure Divisions of the source program it is shown below.  Optimization
which would require modification to the Data and Procedure Divisions is not
considered in this report in that it is beyond the scope of the use of
standard COBOL and the validation process.

    N/A

4.  Compiler.

    IPI BLIS/COBOL  Version 2.0

5.  Operating system.

    IPI BLIS/COBOL Verson 2.0

6.  COBOL reference manual.

    BLIS/COBOL Programmer's Manual  -  Dated  11/20/75

CCVS74-VSR175

APPENDIX A

VALIDATION SUMMARY WORKING DOCUMENT

A-1     This appendix is a working paper produced during the validation
and documents the results of the compilation and execution of each of the
programs comprising the CCVS.  The results contained herein are based on
the use of the compiler within the Validation Environment identified in
this appendix.  This appendix (Validation Summary Working Document) is not
part of the official Validation Summary Report (VSR) and is not intended
to reflect in any way the compiler's usefulness or degree of conformance
to the language specifications.

        The reader of this appendix should keep in mind that the same pro-
blem area may appear in more than one program, but is considered only as one
single discrepancy and as such is reflected only once in the body of the
VSR.  (The VSR will in turn only reference the first occurrence of the
problem in the appendix.)

        This appendix is divided into two parts.  The first part describes
the Validation Environment.  The second part of the document is divided into
categories of information:  compilation and execution results.

        The reference document for COBOL is FIPS PUB 21-1  (X3.23-1974).

VALIDATION ENVIRONMENT

COMPILER IDENTIFICATION:     IPI  BLIS/COBOL   - Version 2.0

COMPUTER SYSTEM:             MM1  NOVA 1200  Compatible

OPERATING SYSTEM:            IPI  BLIS/COBOL   - Version 2.0

NUCLEUS MODULE LEVEL 1

VC101

    A.  Compilation

        1.  Compiler messages of the type

           244C  WRK-DS-06V06 REDEFINES ONLY PERMITTED FOR LEVELS 01-49

           were produced by the compiler on 77 level WORKING-STORAGE
           entries which specified the REDEFINES phrase.

        2.  Compiler messages of the type

           210C  11111  ILLEGAL AREA ENTRY OR MISSING PREVIOUS PERIOD

           were produced when noninteger numeric literals were the
           object of the VALUE IS clause.

        3.  Compiler messages were produced indicating that the PICTURE
           character-string symbol "P" is ignored.  The BLIS/COBOL
           Programmers Manual states that this feature is not supported.

        4.  Compiler messages of the type

           107W  COMPUTATIONA THIS CLAUSE IS IGNORED BY BLIS/COBOL
                VERSION 1.8

           were produced by the compiler on all Data Division entries
           which included USAGE COMPUTATIONAL or USAGE COMP in their
           data descriptions.  The BLIS/COBOL Programmers Manual states
           that this feature is not supported.

    B.  Execution

    The program stopped during execution following the printing of
    the results of MPY-TEST-1.  When the program was terminated at
    the operator's console the following message was displayed

    3760  SUSPEND #2  USER REQ.

This program tests the different forms of the level 1 MULTIPLY and DIVIDE statement.

NC102

    A.  Compilation

        1.  Compiler messages were produced by the compiler on all references to the ALTER verb. The PLIS/COBOL Programmers Manual states that the ALTER statement is not supported.

        2.  Compiler messages were produced on all PERFORM ... identifier TIMES where the data description of the identifier included a "S" symbol in the PICTURE character-string.

    B.  Execution

        1.  All ALTER tests failed due to A.1 above.

        2.  Program execution stopped with the following system message

           11350  SUSPEND #16.

           The test was testing a PERFORM statement which was nested 20 levels deep.

NC103

    A.  Compilation

        All references to switches in the SPECIAL-NAMES paragraph were flagged by the compiler. The PLIS/COBOL Programmers Manual states that the mnemonic-name and condition-name of the SPECIAL-NAMES paragraph are not supported.

    B.  Execution

        The size of the object program was too large to execute on this system. This program tests the different options of the IF statement.

NC104

    A.  Compilation

        Same as NC101 A.3 the PICTURE symbol "P".

    B.  Execution

        1.  MOVE-TEST-6 and MOVE-TEST-25 failed due to NC104 A above. The MOVE statement operands specified the scaling symbol "P".

CCVS74-VSR175

2.  The following tests failed to give the correct answer when
    the sending item was an alphanumeric elementary item and the
    receiving item was a numeric elementary item.

    MOVE-TEST-36.   MOVE PIC X(5) TO PIC 999V99

        Computed   123.45
        Expected   345.00

    MOVE-TEST-37.   MOVE PIC X(5) TO PIC 9(5)V99

        Computed   123.45
        Expected   12345.00

3.  Although no compile-time or execution-time messages were issued
    for data-names and procedure-names which were greater than 12
    characters, only the first 12 characters of the name are used by
    this compiler.

NC105

A.  Compilation

    1.  The compiler produced a message on the PICTURE character-string
        X(26)9X(12)0X(10).  The message stated that "EDITED PIC CHAR-
        STRINGS MAY NOT EXCEED 30 CHARS".

    2.  The compiler produced messages indicating that the RIGHT option
        of the SYNCHRONIZE clause was ignored by BLIS/COBOL Version 2.0.
        The BLIS/COBOL Programmers Manual states that the word RIGHT
        of the SYNCHRONIZED clause is not supported.

    3.  The compiler produced a message indicating that the COMP clause
        was ignored by BLIS/COBOL Version 1.X.  The BLIS/COBOL Programmers
        Manual states that USAGE IS COMPUTATIONAL is not supported.

B.  Execution

    The size of the object program was too large to execute on this
    system.  This program tests different options of the MOVE statement.

NC106

A.  Compilation

    1.  Same as NC101 A.1 thru A.4.

    2.  Between SUB-TEST-6 and SUB-TEST-10 of the source program the
        compiler produced several messages such as

        13960  461C  S6-FAIL-6 COBOL WORD CHARS MUST BE A-Z, 0-9,
                     HYPHEN

34

14040    317C    FRO UNDEFINE DATA-NAME
14070    442C    . FROM MUST TERMINATE SUMMED SERIES (SUBTRACT
                  VERB)
14510    144C    DO NOT A VERB (VERB EXPECTED)

The words identified in the messages were parts words from
the COBOL statement and new words not present in the programs.
The COBOL source lines were testing the SUBTRACT statement
which included the ROUNDED and ON SIZE ERROR phrases.

3.   Compiler messages were produced stating that "PROCEDURE-
     NAME TABLE IS FULL IN COMPILER".

B.   Execution

1.   ADD-TEST-3 thru ADD-TEST-5, ADD-TEST-8 thru ADD-TEST-10,
     ADD-TEST-20 thru ADD-TEST-23, ADD-TEST-26 thru ADD-TEST-27,
     ADD-TEST-39 thru ADD-TEST-42, and ADD-TEST-52 test the
     ON SIZE ERROR phrase of the ADD statement.  In each test
     the ON SIZE ERROR path was not taken when a size error
     condition occurred.  Also in each case the contents of
     the resultant-identifier(s) which should not be affected
     when a size error occurs was altered.

2.   ADD statements of the type ADD operand-1, operand-2 ...
     GIVING identifier in which the operands were a mixture
     of integer and noninteger values did not produce the
     expected answers.

                      Computed                 Expected
                      --------                 --------

     ADD-TEST-15      355888.666333            344777.777443
     ADD-TEST-17      355555.000000            333334.000000

3.   ADD-TEST-30 thru ADD-TEST-32 and ADD-TEST-38 failed because
     the ADD statement operands specified the scaling symbol "P"
     in the PICTURE character-string.  The symbol "P" is not
     supported by the compiler.

4.   SUB-TEST-3, SUB-TEST-17 thru SUB-TEST-20 and SUB-TEST-23 thru
     SUB-TEST-24 test the ON SIZE ERROR phrase of the SUBTRACT
     statement.  In each test the ON SIZE ERROR path was not
     taken when a size error condition occurred.  Also in each
     case the contents of the resultant-identifer(s) which
     should not be affected when a size error occurs was
     altered.

5.   SUBTRACT statements of the type SUBTRACT operand-1,
     operand-2, ... FROM operand-n GIVING ... in which the
     operands were a mixture of integer and noninteger values
     did not produce the expected answer.

35

|  | Computed | Expected |
|--|----------|----------|
| SUB-TEST-10 | 311000.222333 | 322111.111223 |
| SUB-TEST-12 | -355888.666333 | -344777.777443 |

6. SUB-TEST-13 and SUB-TEST-27 thru SUB-TEST-29 failed because the SUBTRACT statement operands specified the scaling symbol "P" in the PICTURE character-string.

7. SUB-TEST-36 thru SUB-TEST-53 were not executed due to NC106 A.3 above.

NC107

A. Compilation

1. Data Division entries of the type

   77  DATA-J PICTURE IS WWWWd.

   were flagged by the compiler as containing an "ILLEGAL PIC CHAR STRING". The character W was defined as an operand of the CURRENCY SIGN IS literal clause in the SPECIAL-NAMES paragraph.

2. Compiler messages were produced on numeric literals of the type 9.999.999/9. The clause DECIMAL POINT IS COMMA was specified in the SPECIAL-NAMES paragraph of the program.

3. The compiler message

   1630  2610 000  NUMERIC ITEMS REQUIRE VALUE LITERAL
        TO BE NUMERIC

   was issued by the compiler on the following Data Division entry

   77  DATA-P PICTURE 999 VALUE "000" BLANK WHEN ZERO.

4. Compiler messages were produced on 77 level entries which included the REDEFINE phrase. See NC101 A.1.

5. The ALTER statement is not supported by this compiler. See NC102 A.1.

6. For procedure-names composed only of numeric digits and greater than 18 characters in length the compiler produced the message "NUMERIC LITERALS MUST BE 1-18 DIGITS IN SIZE".

36

7. The message

06490    415W    SEP-WRITE-1.    REFERENCE TO AN UNDEFINED
                                 PROCEDURE-NAME

was produced by the compiler on a GO TO statement in which
the statement ended in column 72 of the COBOL source line.

8. The message

12580    415W    50    REFERENCE TO AN UNDEFINED PROCEDURE-NAME

was produced by the compiler for the following procedures:

02.
     GO TO 50.
50.    PERFORM FAIL.
     GO TO NUM-WRITE-1.

B. Execution

The program stopped execution when encountering a paragraph in
which all the statements were Right Justified to margin R
(column 72).

NC108

A. Compilation

1. The mnemonic-name and switch-condition names in the SPECIAL-
   NAMES paragraph are not supported by this compiler.

2. Same as NC105 A.2 - RIGHT option of the SYNCHRONIZED clause.

3. A compiler message was produced for the Data Division
   entry:

   02  MORE-COMPLETE-FORMAT
       BLANK WHEN ZERO
       PICTURE IS 9
       SYNCHRONIZED RIGHT
       DISPLAY
           VALUE IS "5".

   The message stated that "NUMERIC ITEMS REQUIRE VALUE LITERAL
   TO BE NUMERIC" even though the data content of the alpha-
   numeric literal was numeric.

4. Same as NC105 A.3 - USAGE IS COMPUTATIONAL clause.

5. Same as NC107 A.1 - CURRENCY SIGN IS literal.

37

6. Program compilation terminated with the system message

    WORKING-STORAGE
    103.644  SUSPEND #3 INVALID OBJECT MODULE FORMAT.

B. Execution

Execution of this program was not attempted due to A.6 above.
This program tests the GO TO and PERFORM statements.

## NC109

A. Compilation

Program compilation terminated with the system message

PROCEDURE DIVSN
127.464  SUSPEND #6 OUT OF RANGE SUBSCRIPT

B. Execution

Execution was not attempted due to NC109 A above.  This program
tests the ACCEPT, STOP, and DISPLAY statements.

## NC110

A. Compilation

Same as NC102 A.1 - The ALTER Statement.

B. Execution

No errors.

## NC111

A. Compilation

Same as NC101 A.3 - PICTURE character-string symbol "P".

B. Execution

The system issued the message

SUSPEND #3 INVALID OBJECT MODULE FORMAT

when an attempt was made to execute the program.  This program
tests various options of the ADD and SUBTRACT statements.

## NC112

A. Compilation

38

No errors.

B.  Execution

No errors.

NC113

A.  Compilation

The compiler produced the message

2100  FILLER ILLEGAL AREA-A ENTRY OR MISSING PREVIOUS PERIOD

on the following Data Division entry

02  FILLER PICTURE X(42) VALUE ...

where the word FILLER started in column 11 of the source line.

B.  Execution

The program terminated with the system message

1170  SUSPEND #16 MAX PERFORM NESTING EXCEEDED.

NC114

A.  Compilation

1.  The compiler message

00590  112W  AMERICAN-IND IMPLEMENTOR-NAME/ALPHABET-NAME
              IS UNDEFINED

was produced on the entry AMERICAN-INDIAN IS NATIVE in the
SPECIAL-NAMES paragraph.  The BLIS/COBOL Programmers Manual
states that the alphabet-name clause is not supported.

2.  Compiler messages were produced on Data Division entries
specifying the LEADING option of the SIGN clause.  The
BLIS/COBOL Programmers Manual states that the LEADING clause
is not supported.  This compiler uses only the SIGN IS
TRAILING SEPARATE CHARACTER form of the SIGN clause for the
position and mode of representation of the operational SIGN.

3.  Same as NC101 A.3 - The PICTURE character-string symbol
"P".

4.  Program compilation terminated with the system message

39

6.060   SUSPEND #3 INVALID OBJECT MODULE FORMAT

B.   Execution

Program execution was not attempted due to NC114 A.4 above.
This program tests the various options of the SIGN and
REDEFINES.

NC115

A.   Compilation

All references to the COBOL INSPECT verb were flagged by the
compiler as not supported.

B.   Execution

Due to NC115 A above execution of this program was not attempted.

NC116

A.   Compilation

This program contains data descriptions which use the
LEADING option of the SIGN clause in its definition.   See
NC114 A.2.

B.   Execution

1.   Tests SIGN-TEST-01.01 thru SIGN-TEST-01.04, MOVE-TEST-03.01
     MOVE-TEST-03.03, MOVE-TEST-05.01, MOVE-TEST-05.02, and
     MOVE-TEST-07.01 thru MOVE-TEST-07.06 failed due to NC116 A
     above.

2.   In IF-TEST-02.01 and IF-TEST-02.03 the comparison of a data
     item described as S9(8) SIGN TRAILING VALUE - 01234567 with
     a data item described as S9(7) COMPUTATIONAL VALUE - 1234567
     and S9(7) DISPLAY VALUE -1234567 respectively did not give
     an equal condition when used in an IF statement.   The only
     form of the SIGN clause supported by this compiler is
     SIGN IS TRAILING SEPARATE CHARACTER.

3.   In MOVE-TEST-05.04 a data item described as S9(4) SIGN IS
     TRAILING is moved to a data item described as S9(5) SIGN
     IS TRAILING SEPARATE.

     Computed   03621D
     Expected   03621-

NC117

A.   Compilation

40

1.  The same type of compiler message on the REDEFINES of 77 level
    entries, noninteger literals in the VALUE clause, PICTURE
    symbol P and USAGE COMPUTATIONAL clause resulted from
    compilation of the program as was noted in NC101 A above.

2.  The same type of compiler messages on the SIGN clause
    resulted from compilation of this program as was noted
    in NC114 A.2 above.

3.  The compiler message

    373C  DIVIDE 18 DIGIT ACCUM DOES NOT PERMIT ROUNDING
          INDICATED

    was produced on the following statement

    DIVIDE A18ONES-DS-09V09 INTO WRK-DS-09V09 ROUNDED ON
    SIZE ERROR ...

B.  Execution

1.  DIV-TEST-8 checked the resultant-identifier (DIV7) following
    execution of the statement

    DIVIDE 0.097 INTO DIV7 ROUNDED ON SIZE ERROR ...

    where a size error condition was expected.  The ON SIZE ERROR
    path was taken as expected, but the contents of DIV7 which
    should not be altered when a size error condition occurs was
    altered.

    Computed Data   9.0
    Expected Data   9.6

2.  Tests DIV-TEST-11 thru DIV-TEST-13 checked the ON SIZE ERROR
    path and contents of the resultant-identifier resulting from
    execution of the statement

    DIVIDE DIV4 INTO DIV2 GIVING DIV10 ON SIZE ERROR...

    and

    DIVIDE 1.0051 INTO 100.50 GIVING DIV8 ROUNDED ON SIZE ERROR...

    The ON SIZE ERROR path was not taken where a size error
    condition was expected and the data contents of the
    resultant-identifiers were altered.

3.  In DIV-TEST-17 and DIV-TEST-18 the ON SIZE ERROR path was
    not taken on a size error condition for the following

statements

DIVIDE DIV2 BY DIV4 GIVING DIV10 ON SIZE ERROR...
DIVIDE 100.50 BY 1.0051 GIVING ROUNDED ON SIZE ERROR...

4.  Tests DIV-TEST-21 thru DIV-TEST-24, DIV-TEST-30, DIV-TEST-35,
    DIV-TEST-40, DIV-TEST-45, and DIV-TEST-50 all failed.  The
    above tests used a combination of identifiers in the DIVIDE
    statement which specified the PICTURE character symbol "P"
    and the SIGN IS LEADING in their Data Descriptions.  See
    NC117 A above.

5.  DIV-TEST-27 failed due to NC117 A.3 above.

## NC118

A.  Compilation

Program compilation terminated with the system message

124.350 SUSPEND #3 INVALID OBJECT MODULE FORMAT

B.  Execution

Due to NC118 A above program execution was not attempted.

## NC119

A.  Compilation

1.  The same type of compiler message on the REDEFINES OF 77 level
    entries, noninteger literals in the VALUE clause, PICTURE
    symbol P and USAGE COMPUTATIONAL clause resulted from com-
    pilation of this program as was noted in NC101 A above.

2.  The same type of compiler message on the SIGN clause resulted
    from compilation of this program as was noted in NC114 A.2
    above.

B.  Execution

1.  In SUB-TEST-3, SUB-TEST-7 and SUB-TEST-53 the SUBTRACT
    statement

SUBTRACT N-10 FROM N-13 ON SIZE ERROR ...

and

SUBTRACT N-25 FROM -99999 GIVING N-26 ON SIZE ERROR ...

and

42

CCVS74-VSR175

SUBTRACT 1 FROM WRK-CS-03V00 ON SIZE ERROR ...

fail to take the ON SIZE ERROR path when a size error
condition occurred.

2.  SUB-TEST-10, SUB-TEST-12, SUB-TEST-13, SUB-TEST-17 thru
    SUB-TEST-20, SUB-TEST-23, SUB-TEST-24, SUB-TEST-27 thru
    SUB-TEST-29, SUB-TEST-36, SUB-TEST-37, and SUB-TEST-39
    thru SUB-TEST-46 failed due to A above.

NC120

A.  Compilation

Same as NC119 A above.

B.  Execution

1.  MPY-TEST-19 and MPY-TEST-20 tested the use of the ON SIZE
    ERROR clause with the MULTIPLY statement.  The ON SIZE
    ERROR path was not taken when a size error condition occurred.
    Also the contents of the resultant-identifier which should
    not be affected when a size error condition occurs was
    altered.

2.  Tests MPY-TEST-9 thru MPY-TEST-12, MPY-TEST-14, MPY-TEST-23,
    MPY-TEST-24, MPY-TEST-27, MPY-TEST-29 thru MPY-TEST-31,
    MPY-TEST-33 thru MPY-TEST-35 failed due to A above.

SEQUENTIAL ACCESS LEVEL 1

GENERAL: (1) Because there was only one magnetic tape drive on the system the files which were to be placed on magnetic tape were assigned to disk.

(2) All references to CLOSE UNIT and CLOSE REEL were deleted from the test programs. The BLIS/COBOL Programmers Manual states that the REEL and UNIT phrases of the CLOSE statement are not supported.

SQ101

A. Compilation

No errors.

B. Execution

In paragraph WRT-TEST-008 and WRT-TEST-016 the statement WRITE ... AFTER 0 LINE is tested. Line overprinting was expected by virtue of the fact that no line advancing (0) was indicated. However, on this system line advancing did occur before the line was printed.

SQ102

A. Compilation

The STANDARD option of the LABEL RECORD clause was flagged by the compiler. The BLIS/COBOL Programmers Manual states that the STANDARD form of this clause is not supported.

B. Execution

No errors.

SQ103

The procedure and data names were not unique within the first 12 positions as required by this compiler, therefore the program was not run.

SQ104

A. Compilation

The maximum physical record size and logical record size is 512 characters (bytes) for this system.

B. Execution

44

No errors.

SQ105

A. Compilation

The compiler flagged the word DECLARATIVES in the Procedure Division. The BLIS/COBOL Programmers Manual states that DECLARATIVE and USE statements are not supported.

B. Execution

When an attempt was made to execute the program the following system message was displayed

SUSPEND #3 INVALID OBJECT MODULE FORMAT

The program creates and processes a file with fixed length records.

SQ106 and SQ107

The procedure and data names were not unique within the first 12 positions as required by this compiler, therefore this program was not run.

SQ108

A. Compilation

No errors were noted on initial program compilation. Subsequent attempts to compile the program resulted in the system message

611.720 SUSPEND #3 INVALID OBJECT MODULE FORMAT

being displayed at the terminal. The program was testing use of the READ ... INTO statement.

B. Execution

Due to A above this program was not run.

SQ109

A. Compilation

No errors.

B. Execution

This program creates a file with 750 records and then uses

different forms of the READ statement to read the file. An
unexpected end-of-file occurred when reading the file with
the statement READ ... RECORD AT END ... . A similar test
in program SQ104 passed.

SQ110

    A.  Compilation

    No errors.

    B.  Execution

    No errors.

SQ111

    A.  Compilation

    This compiler did not accept the CODE-SET IS alphabet-name where
    alphabet-name is defined in the SPECIAL-NAMES paragraph as
    required by the language specifications. Support of the STANDARD-1
    (ASCII) code set is claimed however the word STANDARD-1 must be
    the object of the CODE-SET clause.

    B.  Execution

    Due to A above the CODE-SET test failed.

SQ112

    A.  Compilation

    No errors.

    B.  Execution

    Program execution terminated before normal completion with the
    system message

    3630  SUSPEND #55 VIOLATION, END EXTENT D2 SOURCE 1

SQ113

    A.  Compilation

    This compiler does not support the RERUN clause.

    B.  Execution

    This program uses as input the file produced in SQ112. Due to SQ112
    B and SQ113 A above this program was not executed.

46

SQ114

    A. Compilation

    No errors.

    B. Execution

    All the tests in this program failed. The program was testing use of the SAME AREA clause.

SQ115

    A. Compilation

    No errors.

    B. Execution

    Program execution terminated while executing test REWRITE-TEST-01. This test was testing the REWRITE record-name form of the REWRITE clause.

SQ116

    A. Compilation

    Program compilation terminated before normal completion with the system message

    611.720 SUSPEND #3 INVALID OBJECT MODULE FORMAT.

    This program tests use of the REWRITE ... FROM option of the REWRITE statement.

    B. Execution

    Due to A above this program was not executed.

SQ117

    A. Compilation

    No errors.

    B. Execution

    Program execution terminated before normal completion with the system message

    4640 SUSPEND #55 BOUNDARD VIOLATION END EXTENT 02 SOURCE 2

47

The program was in the process of creating the 51st to 100th
record of the file with the statement WRITE record-name FROM
identifier.  The record size of the identifier was larger
than that of the record-name record.

S0118

   A.  Compilation

   Same as S0111 A above.

   B.  Execution

   No errors.

TABLE HANDLING LEVEL 1

TH101

A. Compilation

This compiler does not support the indexing feature of the
COBOL Table Handling module. All uses of the elements

USAGE IS INDEX
INDEXED BY
SET statement
relative indexing

were flagged by the compiler.

B. Execution

The program was not executed due to A above.

TH102 and TH103

A. Compilation

No errors.

B. Execution

No errors.

TH104

A. Compilation

Same as TH101 A above.

B. Execution

Same as TH101 B above.

TH105

A. Compilaton

1. Compiler messages of the type

05110  33W  10+  SUSCRIPT LIT'RL MUST BE UNSIGNED,
                 NON-ZERO INTGR

were produced on subscripts which were composed of a numeric
integer preceeded by a plus sign.

49

2. The compiler message

05900 073E TO ILLEGAL SYNTAX ROUTINE # IN COMPILER *****

was produced on the following source code.

```
058900     MOVE 1 TO SUBSCRIPT-3
059000     MOVE 1 TO SUBSCRIPT-2.
```

3. The message

04270 415W TTST-41.M REFERENCE TO AN UNDEFINED PROCEDURE-
NAME.

was produced by the compiler. The line 04270 in the above
message is a source line which contains only a procedure-
name which is the object of a GO TO statement. The source
code sequence was

```
042600     IF ... ELSE GO TO
042700     TTST-41.
042800     GO TO ...
042900  TTST-41.
```

B. Execution

The system message

0 SUSPEND #76 NON CURRENT PROG MODULE LOADED

was displayed at the console when an attempt was made to
execute the program.

TH106 thru TH110

A. Compilation

Same as TH101 A above.

B. Execution

Same as TH101 B above.